1972

A Note on Orders of Enumeration

John Helm

Albert Meyer

Paul Young

Report Number:
72-050

ON ORDERS OF ENUMERATIONS AND TRANSLATIONS

by

John Helm[1], Albert Meyer[2], & Paul Young[3]

January 1972

CSD TR 50

PURDUE UNIVERSITY

A basic result of intuitive recursion theory is that a set (of natural numbers) is decidable (recursive) iffit can be effectively enumerated in its natural order (of increasing magnitude). The chief theorems of this paper give simple, but very basic facts relating to enumerations in natural order of magnitude and the extent to which translations must preserve the orders of the sets being enumerated. Under very general conditions for what constitutes a programming system for enumerating the recursively enumerable (r.e.) sets, we prove in Theorem 1 that not every recursive set is "best" enumerated in its natural order, and we later show that under these same general conditions, in every programming system, every recursive set is enumerable in its natural order. We accomplish the latter result by extending Rogers' Isomorphism Theorem to a result which asserts that under these same general conditions, every programming system can be effectively translated into any other in a manner which preserves the order of the sets being enumerated, (Theorems 4 and 5). In fact, we show that for every translator, t, there is an order preserving pretranslator, p, such that t is order preserving modulo p; i.e. $t \circ p$ is an order preserving translator. In addition, the restriction of the original programming system to the recursive set {range p} is a standard programming system on which t is order preserving. Along the way we establish the existence of

sets best enumerated in their natural order and, for every r.e. set, the existence of bad orders for enumerating the set. A companion paper, "Notes on difficulties of enumerations," contains some observations relating difficulty of enumerations, difficulty of computations, difficulty of decision problems, one-one and many-one reducibilities, and classifications of the r.e. sets. All proofs in this paper are fairly straightforward.

NOTATION AND BASIC DEFINITIONS

We let $\lambda x D_x$ denote a canonical enumeration of all finite sets (of natural numbers). Given x, we can effectively list D_x and know when the listing is completed. Except for the indexings $\lambda x W_x$ of the r.e. sets which we are about to describe and the notation W_x^n described later, our notation generally follows [7]. $\langle x,y \rangle$ is (an encoding of) the ordered pair of integers x and y.

Definition, ([8]). An enumeration technique is a total recursive function E(x,y) such that for every (r.e.) set W there exists an e such that $W = \underset{n}{\cup} D_{E(e,n)}$. We call e an index of W and denote $\underset{n}{\cup} D_{E(e,n)}$ by W_e^E, the superscript E being suppressed whenever there is no danger of ambiguity. $D_{E'(e,n)} = \underset{m \leq n}{\cup} D_{E(e,m)}$.

An enumeration technique is called standard if the S_1^1- theorem is satisfied, i.e., if there exists a total recursive function S such that for all e and x, $W_{S(e,x)} = \{y \mid \langle y,x \rangle \in W_e\}$. We assume without further mention that all enumeration techniques considered in this paper are standard.

The reader should have no trouble convincing himself that all of the usual models for enumerating r.e. sets obviously yield standard enumeration techniques. E.g., we might let $D_{E(e,n)}$ denote the set of integers enumerated either in exactly n steps of Turing machine e or in no more than n steps of Turing machine e. Clearly, each enumeration technique E partially orders each r.e. set W_e^E in a natural way which can be extended to a total ordering; that is, we may define $x <_e^E y$ if x precedes y in the enumeration of W_e^E, and in the case that x and y appear at the same stage of the enumeration we assume that x and y are arbitrarily ordered by some convention uniform in e:

Definition. For each enumeration technique E, $x <_e^E y$ is some fixed r.e. ternary relation of x, e, and y such that for each integer e,

(i) $<_e^E$ totally orders W_e,

and (ii) $[p > o \;\&\; x \in D_{E'(e,n)} - D_{E'(e,n-1)} \;\&\; y \in D_{E'(e,n+p)} - D_{E(e,n+p-1)}]$

$\Rightarrow [x <_e^E y]$.

Clearly, if the binary relations $<_e^E$ and $<_{e_o}^{E_o}$ are identical, $W_e^E = W_{e_o}^{E_o}$. We shall generally be working with two arbitrary, but fixed enumeration techniques E and E_o. We abbreviate W_e^E and $W_e^{E_o}$ to W_e and W_e^o respectively and similarly $<^E$ and $<^{E_o}$ by $<$ and $<^o$.

Definition, ([8]). For each enumeration technique E, we define $A_j(n) = (\mu y)[|D_{E'(j,y)}| > n]$. If $A_j(n)$ exists, W_j^n is the set consisting of the first n elements of W_j in the order $<_j$. This is similar to

Definition, ([1]). For each standard indexing $\lambda i \underline{\phi}_i$ of the partial recursive functions, a Blum measure is a r.e. sequence of partial recursive functions $\lambda i \underline{\Phi}_i$ such that for all i, domain $\underline{\phi}_i$ = domain $\underline{\Phi}_i$ and the ternary relation $\underline{\Phi}_i(x) \leq y$ is decidable (under the convention that it is false if $\underline{\Phi}_i(x)$ is undefined).

I ORDERS OF ENUMERATIONS

We now prove that any infinite, coinfinite, recursive set has a recursive superset whose natural order is not the best order for enumerating the set. The original proof, [2], was by priority argument, but the proof given here is more in the spirit of [3], and seems to bear evidence for the claim made in [3] that "the existence of almost everywhere complex zero-one valued functions is often sufficient to establish the existence of inherently complex computations".

Although the proof we are about to give is largely calculational, the reader will be well-advised to bear in mind that the calculations are merely formal validation of the following intuitive argument: Let R be any infinite and coinfinite recursive set, and let f be a 0-1 valued function which is much harder to compute than R is to enumerate in increasing order. Let $\overline{r}$ enumerate $\overline{R}$ in increasing order and let $S = R \cup \{\overline{r}(n) \mid f(n) = 0\}$. Since $R \subseteq S$, S can be enumerated almost as easily as R, ([10]). But if S can be enumerated easily in natural order, then S-R can also be enumerated easily in natural order. But if this were possible, the following algorithm would give an easy way to compute f(n): enumerate the first n+1 elements of S-R; if $\overline{r}(n)$ appears set f(n) = 0, otherwise f(n) = 1. We now formalize this in:

Theorem 1. Let R be an infinite and coinfinite recursive set. Let (F) be any effective operator carrying total recursive functions to total recursive functions. Then there exists a recursive set S such that $R \subseteq S$, and some index e for S such that for any j if $W_j = S$ and $<_j$ is the increasing order for S,[4] then $(F(A_e))(n) \leq A_j(n)$ for all but finitely many n.

Proof. Let $\overline{r}$ enumerate $\overline{R}$ in 1-1 increasing order. Define $\overline{r}^{-1}$ by $\overline{r}^{-1}(y) = \max\{z \mid \overline{r}(z) \leq y\}$. (Thus $\overline{r}\,\overline{r}^{-1}(y) \leq y$ and $\overline{r}^{-1}(y-1) \leq y$.) Define a total recursive function ℓ by

$$W_{\ell(i)} = R \cup \{\overline{r}(n) \mid \phi_i(n) = 0\},$$

and define a very large upper bound, t, on the difficulty of enumerating these supersets of R by

$$t(z) = \max_{i \leq z} \{(F(A_{\ell(i)}))(y) \mid \overline{r}^{-1}(y-1) = z\}.$$

Thus for any i, for almost all y, (a.e., y)

(a) $\quad (F(A_{\ell(i)}))(y) \leq t(\bar{r}^{-1}(y-1)).$

Define

$$\underline{\phi}_{s(e)}(y)\begin{cases} = \text{undefined if } W_e \text{ has fewer than } \bar{r}(y)+1 \text{ elements} \\ = 0 \quad \text{if } \bar{r}(y) \in W_e^{\bar{r}(y)+1} \\ = 1 \quad \text{otherwise.} \end{cases}$$

Define

$$S_o(i,n,y)\begin{cases} = \underline{\phi}_{s(i)}(n) \quad \text{if } A_i(r(n)+1) \leq y \\ = 0 \quad \text{otherwise,} \end{cases}$$

and $\quad S(n,y) = \max_{i\leq n}\{S_o(i,n,y), S(n,y-1), S(n-1,y)\},$

so that S is monotone in both variables and enables us to bound the difficulty of computing $\underline{\phi}_{s(i)}$ in terms of the difficulty of enumerations W_i:

(b) $\quad \underline{\phi}_{s(i)}(n) \leq S(n, A_i(\bar{r}(n+1))) \qquad$ a.e., n.

We now choose a characteristic function f which is much more difficult to compute than any of the supersets $W_{\ell(x)}$ of R are to enumerate. Specifically choose a 0-1 valued total recursive function f such that $\underline{\phi}_c = f$ implies

(c) $\quad S(\bar{r}(y+1), t(y)) \leq \underline{\phi}_c(y) \qquad$ a.e.; [4], [2].

Next note that $\underline{\phi}_{s(e)}$ is so defined that <u>if</u> e enumerates $W_{\ell(c)}$ <u>in its natural order</u>,

then $\quad \underline{\phi}_{s(e)} = f(=\underline{\phi}_c).$

Therefore, by (c), $\quad S(\bar{r}(y)+1, t(y)) \leq \underline{\phi}_{s(e)}(y) \qquad$ a.e.

So $\quad S(y, t(\bar{r}^{-1}(y-1))) \leq S(\bar{r}(\bar{r}^{-1}(y-1)+1), t(\bar{r}^{-1}(y-1))) \qquad$ a.e.

$$\leq \underline{\phi}_{s(e)}(\bar{r}^{-1}(y-1)), \qquad \text{a.e.}$$

By (b), $S(y,t(\overline{r}^{-1}(y-1))) \leq S(\overline{r}^{-1}(y-1),A_e(\overline{r}(\overline{r}^{-1}(y-1)+1))$ a.e.

$\leq S(y,A_e(y))$ a.e.

Therefore $t(\overline{r}^{-1}(y-1)) \leq A_e(y)$ a.e.

Finally, by (a), $(F(A_{\ell(c)}))(y) \leq t(\overline{r}^{-1}(y-1)) \leq A_e(y)$ a.e.

Although the preceding theorem asserts that in every enumeration technique, not every recursive set is best enumerated in its natural order, it is clear that for some models some recursive sets are best enumerated in their natural order. E.g., in Turing machine tape measure, the set N of natural numbers and the set of primes are each best enumerated in natural order of magnitude. In fact, although we omit a proof, for any function f, if f(n) is always much bigger than the difficulty of computing f(0), ...f(n-1), then the range of f is best enumerated in its natural order. This motivates the following proof that every enumeration technique has sets which are best enumerated in natural order. We first state a lemma.

Lemma 1. For every enumeration technique E, there exists a total recursive monotone and unbounded function S such that for almost all y,

$$y \in D_{E'(i,z)} \text{ implies } z \geq S(i,y).$$

(I.e., large results cannot be enumerated quickly by a uniform procedure, i.)

Proof. This follows from the recursive relatedness of measures. Specifically, it is certainly true of any of the usual enumeration techniques, so in particular let E_0 be enumeration by Markov algorithms and S_0

the corresponding function for Markov algorithms and assume without loss of generality that $W_x^E = W_x^{E_o}$ for all x.

Define

$$H_o(i,y,n)\begin{cases} = (\mu z)[y \in D_{E'_o}(i,z) & \text{if } y \in D_{E'}(i,n)] \\ = 0 & \text{otherwise,} \end{cases}$$

and $$H_y(n) = \max_{i \le y}\{H_o(i,y,n), H_y(n-1) + 1\}.$$

Now $$y \in D_{E'}(i,n) \Rightarrow y \in D_{E'_o}(i,H_o(i,y,n))$$

$$\Rightarrow y \in D_{E'_o}(i,H_y(n)) \qquad \text{for } y \ge i$$

$$\Rightarrow H_y(n) \ge S_o(i,y)$$

$$\Rightarrow n \ge H_y^{-1}(S_o(i,y)) \qquad \text{for } y \ge i.$$

We thus set $S(i,y) = H_y^{-1}(S_o(i,y))$.

Theorem 2. Let E be any enumeration technique and r any total recursive function. There exists a recursive set R with index e_o such that e_o enumerates R in natural order but for any j, for any $n > j$, if z is in R and if z is one of the first n elements of W_j (in the order $<_j$) but is not one of the first n elements of W_{e_o}, then $r(A_{e_o}(n),n) < A_j(n)$. (Clearly R is best enumerated in its natural order.)

Proof. Define a total recursive function f by defining $W_{f(e)}$ in Stages. At Stage 0, place 0 into $W_{f(e)}$ and at Stage n ($n \ge 1$) do the following: begin computing $A_e(n)$; if $A_e(n)$ is defined, place into $W_{f(e)}$ the first integer, y, such that y is greater than any of the first n elements of W_e and y is sufficiently large that for all $j \le n$

$$y \in D_{E'}(j,z) \quad \text{implies} \quad z \ge r(A_e(n),n).$$

A quick induction now shows that if we use the recursion theorem to obtain $W_{e_o} = W_{f(e_o)}$, then W_{e_o} is infinite, and if $y \in W_{e_o}$ but y is not one of the first n elements of W_{e_o} in the ordering $<_{e_o}$ then y is greater than any of the first n elements of W_{e_o}. Thus W_{e_o} is an infinite recursive set and e_o must enumerate W_{e_o} in increasing order of magnitude.

Furthermore, for any j, for any $n \geq j$, if $y \in W_{e_o}$ but y is not one of the first n elements of W_{e_o}, then we put y into W_{e_o} only if $y \in D_{E'(j,z)}$ implies $z \geq r(A_{e_o}(n),n)$. Thus if y is one of the first n elements of j,

$$A_j(n) \geq (\mu z)[|D_{E'(j,z)}| \geq n]$$

$$\geq (\mu z)[\, y \in D_{E'(j,z)}] \geq r(A_{e_o}(n),n).$$

It is perhaps worth pointing out that Theorem 2 cannot be extended by replacing the arbitrary recursive function r by any very large general recursive operator. To see that this is so, one first defines a total recursive function $\underline{\sigma}$ such that if W_j has at least $2n$ elements then $W_{\underline{\sigma}(j)}$ has at least n elements and if W_j is infinite $W_{\underline{\sigma}(j)} = W_j$ but $<_{\underline{\sigma}(j)} \neq <_j$. ($W_{\underline{\sigma}(j)}$ is enumerated by "watching" the elements of W_j as they appear and whenever enough have appeared "scrambling" some of them.) Next define a total recursive function h by $h(j,n,y) = A_{\underline{\sigma}(j)}(n)$ if $y = A_j(2n)$, while $h(j,n,y) = 0$ otherwise. Finally define a general recursive operator $\mathcal{O}$ mapping functions, t, to functions by $\mathcal{O}(t)(n) = 1 + \max_{j \leq n} h(j,n,t(2n))$. Now notice that for any infinite r.e. set W_i with order $<_i$, $<_i \neq <_{\underline{\sigma}(i)}$, $W_{\underline{\sigma}(i)} = W_i$ and for all $n \geq i$, $\mathcal{O}(A_i)(n) > \max_{j \leq n} h(j,n,A_i(2n)) \geq h(i,n,A_i(2n)) = A_{\underline{\sigma}(i)}(n)$. Thus the order $<_i$ is not "$\mathcal{O}$ better" than the order $<_{\underline{\sigma}(i)}$.

Theorem 2 shows not only that some recursive sets have a best order but also that some orders are (infinitely often) worse than this best order. It is known (although difficult to show) that for some recursively enumerable sets, for every order of enumerating the set there is a much better order, [9]. Whether there are recursive sets with this property appears to be a difficult technical question. On the other hand a simple diagonalization shows that every r.e. set has very bad orders for enumerating the set:

Theorem 3. Given any index e for an infinite r.e. set S and given any total effective operator Φ, there is an index j for S such that for all k, if $W_k = S$ and $<_k = <_j$, then $A_k > \Phi(A_e)$ a.e.

Proof. The index j is determined by the following set of instructions for enumerating a set B. (We shall prove that B = S.) Let $\lambda i s_i$ be some fixed effective enumeration of S. To enumerate B at State n ($n \geq 0$):

1. Find the least p (if any) such that $2p \leq n$, p is not cancelled and $A_p(n) \leq \Phi(A_e)(n)$. If such a p is found, let q denote the n^{th} member of W_p. Let r be the least number such that $s_r \neq q$ and s_r is not yet in B. Add s_r to B and cancel p. 2. If no such p is found, add s_r to B where r is the least y such that s_y has not been added to B.

To see that the construction does what is claimed, we note that the intuitive description given above enumerates the set B in a certain order and that we can build a Turing machine which would enumerate B in this very

same order. Therefore, by Theorem 4 which follows, there is an index j in the enumeration technique under consideration which enumerates B in the order of the above intuitive enumeration. If $A_p(n) \leq (\Phi A_e)(n)$ infinitely often, then it is clear that p must be cancelled, and the intuitive description then makes clear that $<_j \neq <_p$. It now suffices to prove that B = S. Clearly $B \subseteq S$, and since Clause 1 can obtain at most n/2 times prior to State n, Clause 2 must apply infinitely often, forcing B = S.

II TRANSLATIONS

The proof of Theorem 3 used the fact that if a set is enumerable in a given order by Turing machines, it is enumerable in the same order in any enumeration technique. We verify this in Theorems 4 and 5 which extend Rogers' Isomorphism Theorem to an order isomorphism theorem. The proofs simply extend Rogers' proofs. We first state a well-known (translation) lemma which we then generalize in Theorem 4. We also use a generalization of the recursion theorem.

Lemma 2. (Translation; [6]) Given enumeration techniques E and E_o, there exists a total recursive function t such that for all x, $W_x = W^o_{t(x)}$.

Proof. $U = \{\langle y,x\rangle \mid y \in W_x\}$ is r.e. so $U = W^o_p$ for some fixed integer p. But t is then just the total recursive function (guaranteed by the S^1_1-theorem for the indexing $\lambda x W^o_x$) such that $W^o_{t(x)} = \{y \mid \langle y,x\rangle \in W^o_p\}$.

Lemma 3. (Extended Recursion Theorem) Let f(x,y) be a total recursive function and E and E_o arbitrary enumeration techniques. We can effectively find a one-one total recursive function X with recursive range such that for all y, $<^o_{f(X(y),y)} = <_{X(y)}$.

Proof. The methods of the following terse proof are spelled out more fully in the proof of Theorem 4. For fixed x and y, let $i = \langle x,y \rangle$,

$x_0, x_1, x_2, \ldots$ be the members of W_x in the order $<_x$

and $a_0, a_1, a_2, \ldots$ be the members of $W_{f(x,y)}$ in the order $<_{f(x,y)}$.

Define the (noncomputable) function $\underline{\alpha}: N \to N \cup \{\infty\}$ by $\underline{\alpha}(i) = \max\{z \mid a_z$ and x_z are both defined and $a_z = x_z\}$, and define the total recursive function f' by $W_{f'(x,y)} = \{a_0, a_1, \ldots, a_{\underline{\alpha}(i)}, a_{\underline{\alpha}(i)+1}\}$.

It is easy to verify that any total recursive function X satisfying $W_{f'(X(y),y)} = W_{X(y)}$ must also satisfy $<^{o}_{f(X(y),y)} = <_{X(y)}$. The ordinary recursion theorem guarantees the existence of such one-one functions, X, with recursive range.

Lemma 3 now enables us to prove a theorem on translations which has interesting interpretations. Suppose we wish to translate from one programming system into another. The most obvious way would appear to be to use the programs in the image system to simulate the object programs of which they are the translations. Of course, for a few object programs one may be able to recognize from the program exactly what set the program enumerates and hence translate this program without simulating it, and hence change the orders of enumeration. Similarly if the object program enumerates elements very rapidly, one may be able to have the image program enumerate slowly and "watch" the object program to scramble the order in which the object program enumerates elements. Still, these seem ad hoc methods and one might conjecture that the only general method of translation is simulation. Without saying what it means for one program to simulate another, it does seem clear that if program p_0 simulates program p_1 then p_0 and p_1 should enumerate the same

set in the same order.

Our next theorem guarantees that every translator t is, modulo some order-preserving pretranslator X, an order-preserving translator. Alternatively, it guarantees that if we are given enumeration techniques E and E_o with a translation t from E to E_o, then there is a recursive set {range X} such that the restriction of the system E to those programs in {range X} is a standard enumeration technique E', and the fixed translation t translates E' to E in an order-preserving way. I.e., t is order-preserving modulo some recursive set of programs, $\overline{\{\text{range } X\}}$, which are not translated by simulation. Finally the theorem guarantees that every two enumerations techniques E and E_o can be translated into each other in an order-preserving fashion.

Theorem 4. (Order-Translation). Given enumeration techniques E and E_o, and a translation t from E to E_o, we can effectively find a one-one total recursive function X with recursive range such that if we define G(y) to be t(X(y)), then G is an order translation: for all y, $<_y = <^o_{G(y)}$ (and so $W_y = W^o_{G(y)}$). Furthermore X can be so chosen that G(y) always enumerates the elements of $W^o_{G(y)}$ one element at a time; i.e., for all n, $D_{E'_o}(G(y),n+1) - D_{E'_o}(G(y),n)$ has at most one element. Finally, X can be so chosen that $<_y = <_{X(y)}$ for all y.

Proof. Let $y_0, y_1, y_2, \ldots$ the members of W_y in the order $<_y$ and let $X_0, X_1, X_2, \ldots$ be obtained from the sequence $\lambda n[D_{E'_o}(t(x),n+1) - D_{E'_o}(t(x),n)]$ be deleting all empty sets. (So $X_0, X_1, X_2, \ldots$ is the natural partial ordering of W_x by t(x); so it induces a partial suborder of $<^o_{t(x)}$.) We now define a total recursive function $f_o(x,y,n)$ as follows:

$f_o(x,y,o) = y_o$

$f_o(x,y,n+1)$ is undefined unless $f_o(x,y,n)$, y_n, Y_{n+1}, and X_n are all defined and $\{y_n\} = X_n$.

$f_o(x,y,n+1) = y_{n+1}$ if $f_o(x,y,n)$, y_n, y_{n+1}, and X_n are all defined and $X_n = \{y_n\}$.

For each x and y, $\underline{\lambda}nf_o(x,y,n)$ is a computable function and by elementary construction there exists a total recursive function f such that for each x and y f(x,y) is (the Godel number of) a Turing machine which enumerates the range of $\underline{\lambda}nf_o(x,y,n)$ in the order $<^T_{f(x,y)}$ $(= f_o(x,y,0), f_o(x,y,1), f_o(x,y,2),\ldots)$. By our extended Recursion Theorem, there exists a one-one total recursive function X such that $<^T_{f(X(y),y)} = <_{X(y)}$ for all y.

It is clear from the construction that if range $\underline{\lambda}nf_o(X(y), y,n)$ is infinite, then we must have both $<_y = <^o_{t(X(y))}$ and $<_y = <^T_{f(X(y),y)}$ $(= <_{X(y)})$, completing the proof. On the other hand if y_n is the last element placed into range $\lambda nf_o(x,y,n)$, there are three possibilities to consider. If y_{n+1} does not exist, then $W_{X(y)} = W^T_{f(X(y),y)} = \{0,1,\ldots y_n\} = W_y$; since for $m < n$, we must have had $X_m = \{y_m\}$, the only way to have $W_{t(X(y))} = \{0,1,\ldots y_{n-1},y_n\}$ is to also have $X_n = \{y_n\}$, establishing $<^o_{t(X(y))} = <_y$; since it is clear from the construction that if y_{n+1} does not exist, we must have $<^T_{f(X(y),y)} = <_y$ and since $<^T_{f(X(y),y)} = <_{X(y)}$, this again completes the proof. If y_{n+1} exists but X_n is not defined, since $X_m = \{y_m\}$ for all $m < n$, we would have that $W^o_{t(X(y))} = W_{X(y)}$ would have only n elements while $W_{F(X(y),y)}$ has n+1 elements, contradicting $W_{f(X(y),y)} = W_{X(y)}$. Finally if y_{n+1} and X_n both exist but $X_n \neq \{y_n\}$, then X_n contains some elements other than y_n, and since for $m < n$,

$X_m = \{y_m\}$, X_n contains some element not in $\{y_o,\ldots,y_n\} = W_{f(X(y),y)}$; this again contradicts $W_{f(X(y),y)} = W_{X(y)}$.

In closing, we remark that the above proof is easily modified to avoid an appeal to Turing machines and to the recursion theorem, provided we do not require $<_{X(y)} = <_y$.

Corollary 1. In every enumeration technique E, a set is recursive iff it can be enumerated in natural order of magnitude.

Proof. In the system E_o obtained by enumerating sets as ranges of total recursive functions, the result is trivial. It thus follows from Theorem 4.

To give our extension of Rogers' Theorem, we first extend his padding lemma:

Lemma 3. (Order Padding) Let E be any enumeration technique and let e(x) be the least member of D_x. (e(x) = 0 if $D_x = \emptyset$.) There exists a total recursive function p such that if $D_x \neq \emptyset$ and if $y \in D_x$ implies $W_y = W_{e(x)}$, then

$$p(x) \notin D_x \;\&\; <_{p(x)} = <_{e(x)} \quad (\&\; W_{p(x)} = W_{e(x)}).$$

Proof. A direct proof which simply extends the ordinary proof of padding implicit in [6] is a straightforward but slightly tedious exercise. However Lemma 3 may also be proven by observing that it is well-known that there are one-one translations between any two standard indexings. Theorem 4 now guarantees that the one-one translations can be made into one-one order-preserving translations. Since Lemma 3 is clearly true for any of the well-known enumeration techniques such as Turing machines, it follows directly by translating the problem from E to Turing machines, obtaining the new index for Turing machines, and then translating back to E.

Having the Translation Theorem (Theorem 4) and the Padding Lemma (Lemma 3) we can now extend Rogers' Isomorphism Theorem via the usual proof which we merely indicate.

Theorem 5. For any two enumeration techniques E and E_o, there is a recursive permutation t such that $<_e = <^o_{t(e)}$.

Proof. The function t is constructed in Stages. Stage 2n is used to guarantee that the index n appears in the domain of t: The Translation Theorem (from E to E') guarantees that we can find an index n' such that $<_n = <^o_{n'}$, while the Padding Lemma guarantees that we can find an n'' not yet in the range of t such that $<^o_{n'} = <^o_{n''}$, so we define t(n) = n''. Stage 2n + 1 is used to guarantee that t is onto by placing n in the range of t: first the Translation Theorem (from E_o to E) is used to find an index i such that $<_i = <^o_n$ while the Padding Lemma (for E) is then used to find an index i' such that $<_i = <_{i'}$ while i' is not yet in the domain of t so that we may set t(i') = n.

Given two enumeration techniques E and E_o, it is easy to construct translations t from E to E_o ($W_i = W^o_{t(i)}$ for all i) such that for infinitely many distinct r.e. sets W there are indices i of W such that $<_i \neq <^o_{t(i)}$. Nevertheless, we conjecture that every translation must preserve "many" orders. For example, we conjecture that for every translation t there are infinitely many r.e. sets W (of cardinality greater than 1) such that $W_i = W$ implies $<_i = <^o_{t(i)}$. As an immediate consequence of Theorem 4, we know that every translation preserves every order infinitely often:

Corollary 1. Let E and E_o be enumeration techniques and t a translation from E to E_o. Then the set S_j defined by $S_j = \{i \mid <_i = <_j \text{ and } <_i = <^o_{t(i)}\}$ is not recursive. (It is clearly co-r.e.)

Proof. If W_j has at most one element, the Corollary reduces to the well-known fact that $\{i \mid W_i = W_j\}$ is not recursive. Otherwise Theorem 4 guarantees the existence of an e_o such that $W^o_{e_o} = W_j$ but $<_j \neq <^o_{e_o}$. Hence if S were recursive we could obtain a new translation t' for which $\{i \mid <_i = <_j$ and $<_i = <^o_{t'(i)}\}$ would be empty. It thus suffices to prove that S is nonempty. But this also follows immediately from Theorem 4.

FOOTNOTES

1, 2, 3. Partially supported by NSF Research Grant GP-6120, Purdue University[1], Project MAC, an MIT Research program sponsored by the Advanced Research Projects Agency, Dept. of Defense, Office of Naval Research Contract Number Nonr. 4102(01),[2] and NSF Research Grant GJ-27127, Purdue University.[3] Reproduction in whole or in part is permitted for any purpose of the United States Government.

4. It is well-known for any of the usual enumeration techniques that just a j exists. For an arbitrary enumeration technique, the existence of such a j follows from this fact together with our Theorems 4 and 5.

REFERENCES

[1]. Blum, Manuel, A machine independent theory of the complexity of recursive functions, J. Assoc. Comp. Mach., 14 (1967), 332-336.

[2]. Helm, John, Two Topics in Recursion Theory, Purdue Univ. Dissertation, 1970.

[3]. Meyer, A. and McCreight, E., Computationally complex and pseudo-random zero-one valued functions,

[4]. Rabin, M.O., Degree of difficulty of computing a function, Hebrew Univ. Tech. Report 2 (1960).

[5]. Ritter, William, Notation systems and an effective fixed point property, Proc. A.M.S., 17 (1966), 390-395.

[6]. Rogers, Hartley, Godel numberings of partial recursive functions, J. Symbolic Logic 23 (1958), 331-341.

[7]. Rogers, Hartley, Theory of Recursive Functions and Effective Computability, McGraw-Hill, New York, 1967.

[8]. Young, Paul, Toward a theory of enumerations, J. Assoc. Comp. Mach., 16 (1969), 328-348.

[9]. Young, Paul, Speed-ups by changing the order in which sets are enumerated, Math. Systems Theory, 5 (1971), 148 - 156.

[10]. Young, Paul, Speed-ups by changing the order in which sets are enumerated, preliminary abstract, Proc. 1st Annual ACM Symposium on Theory of Computing, ACM, New York (1969), 89-92.